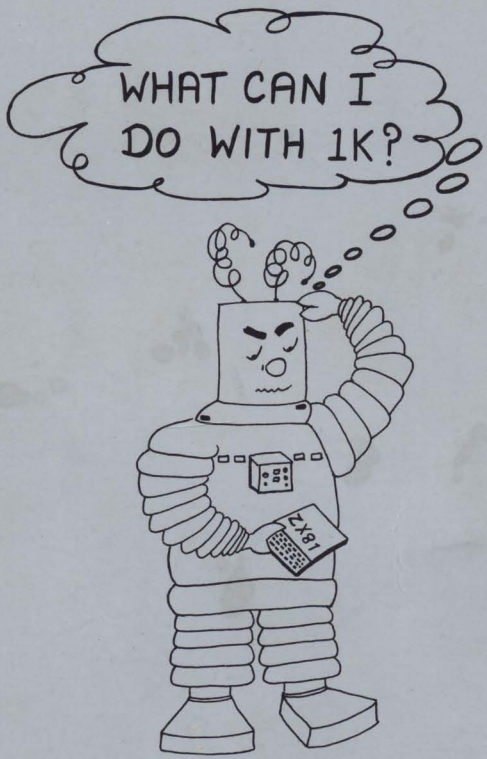


40 PROGRAMS & ROUTINES FOR THE 1K SINCLAIR ZX81

WHAT CAN I DO WITH 1K ?

BY ROGER VALENTINE

40 PROGRAMS & ROUTINES FOR THE UNEXPANDED SINCLAIR ZX81

PUBLISHED BY:
V&H COMPUTER SERVICES
182C KINGSTON ROAD
STAINES
MIDDX.

TEL: STAINES 58041

First Edition January 1982
Second Edition March 1982

© V&H 1982

ISBN 0 946008 00 0

A CASSETTE CONTAINING ALL THE PROGRAMS AND ROUTINES IN THIS BOOK IS AVAILABLE FROM THE PUBLISHERS.

PRICE 4.95

CONTENTS

	Text	Listing
Introduction	5	
Listing Conventions	6	
FATE & FORTUNE		
Numerology	8	9
I Ching	10	11
Chinese Horoscope	12	13
Biorhythm	14	15
Biochart	16	17
Decisions	18	19
PRINTING WITH MORE FRILLS		
Ad.	20	21
Inverse Print	22	23
Demo	24	25
Data Print	26	28
CASINO GAMBLING		
Roulette	29	31
Martingale	30	32
Reverse Martingale	30	32
D.Alambert	30	32
Labouchere	30	32
Result Analysis	33	
Boule	34	35
Noir et Blanc	36	37
Playing Boards	38	
Craps: Dice	40	41
Crap Shooter	42	43
DATA FILES		
Top Ten (Chart Data)	44	45
Computer Dating	46	47
Check Digit	48	49

	Text	Listing
BUSINESS PROGRAMS		
V.A.T.	50	51
Bank Rec.	52	53
Credit Cards	54	55
Payroll	56	58
Test Data	60	
Payslip	61	62
UTILITIES		
Byte Counting	64	65
Line 0	64	65
Auto-Run	66	65
Chains	66	65
Menu	66	67
Machine Code Loader	68	69
Line Renumbering	70	71
M/C Renumbering	72	73
GAMES		
Play Your Cards	74	75
Headlines	76	77
Royalty Cheque	78	79
Bowls	80	81
Catch	82	83
Space Rescue	84	85
Bibliography	86	
Computer Dating Questionnaire	87	

What Can I Do With 1K ?

Many people would answer, Expand. Buy a 16K RAM pack and start running some 'sensible' programs. Our contention, however, is that you should certainly NOT expand until you have fully explored the potential of the standard 1K. In this book, we hope to provide you with some sensible programs which will run in 1K, and hopefully encourage you to write more.

Bad news for some of you: despite an embarrassingly long section on casino gambling, we have not included too many games! In general we have concentrated on the more serious uses for a personal computer. Nevertheless we hope that you will find most of the programs fun to run, (some people enjoy running payrolls!) and possibly pick up a few tips along the way.

N.B. The documentation of some of the programs is deliberately rather limited. This isn't laziness on our part: if you really want to learn about computing, don't ask for everything to be spelled out for you - dive in and work out for yourself just how things work.

LISTING CONVENTIONS

The following conventions have been used throughout this book. Please study them carefully before keying in the programs.

I	=	The letter I
1	=	The number one
O	=	The letter O
0	=	The number zero
'	=	The image quote(SHIFT 0)
←	=	Space (where essential)
↑	=	All items between two up-arrows are to be entered in GRAPHICS
UNDERLINING	=	All underlined items are to be entered in INVERSE

N.B. These conventions have NOT been followed in Line 10 of DATA PRINT, or in DICE.

The conventions for DATA PRINT have been noted in the text.
For DICE, the a graphic square has been used.
(USE Graphic A, H, or Space, as you wish.)

EXAMPLES

```
10 PRINT "↑T↑/FIRST/←EXAMPLE↑Y↑"
```

Enter the string as:

LISTING	KEYSTROKE
↑	SHIFT 9 (graphics ON)
T	SHIFT T (graphic quarter square)
↑	SHIFT 9 (graphics OFF)
/	SHIFT Q
F	F
I	I
R	R
S	S
T	T
/	SHIFT Q
←	SPACE
E	SHIFT 9 (inverse ON)
X	E
A	X
M	A
P	M
L	P
E	L
↑	E
	NO KEYSTROKE ! actually:
	SHIFT 9 (inverse OFF)
	SHIFT 9 (graphics ON)
Y	SHIFT Y (graphic quarter square)
↑	SHIFT 9 (graphics OFF)

Also:

```
10 PRINT "        "
```

is not very helpful, so would be shown as:

```
10 PRINT "←←←←←←←←"
```

Fate and Fortune

No apologies for starting with this section. We know that logical-minded computer people are not supposed to be 'into' this kind of thing, but WE are, and we suspect that secretly you may be too.

N.B. Most of these programs provide data only. You must consult a book on the relevant subject for an interpretation of that data.

Numerology

This program converts any name into a single digit number, according to the rule A=1, B=2, Z=26=2+6=8 etc. The number produced can be taken as a 'lucky number', but is supposed to be indicative of your personality type. Technically you should use the name given to you at birth, but if you also try your nickname, or name by which you are usually known, you can see how you have subconsciously attempted to alter your personality over the years.

NUMEROLOGY

```
5 PRINT "ENTER YOUR NAME"
10 LET N=0
15 INPUT N$
20 CLS
25 IF N$=" STOP " THEN STOP
30 PRINT AT 3,0;"NAME"
35 PRINT AT 5,0;N$
40 FOR J=1 TO LEN N$
45 LET C=CODE N$(J)-37
50 IF C<1 OR C>26 THEN GOTO 80
55 IF C<10 THEN GOTO 70
60 LET C=C-9
65 GOTO 55
70 LET N=N+C
75 IF N>9 THEN LET N=N-9
80 NEXT J
85 PRINT AT 7,0;"NUMBER"
90 PRINT AT 9,0;N
95 PRINT AT 11,0;"NEXT ONE PLEASE"
100 PRINT
105 RUN
```

I Ching

Just run the program to come up with your hexagram for the day. For demonstration purposes, you can continue to produce new hexagrams indefinitely by pressing NEWLINE. However, serious users should run the program just once per day, so delete the last 4 lines if you do not want the repeat facility.

I CHING

```
5 RAND
10 LET P=6
15 LET B$="⟷⟷⟷"
20 FOR K=0 TO 5
25 LET A$="⟷"
30 IF RND<.5 THEN LET A$="⟷"
35 PRINT AT P,11;B$;A$;B$
40 LET P=P+2
45 NEXT K
50 INPUT Q$
55 IF Q$=" STOP " THEN STOP
60 CLS
65 RUN
```

Chinese Horoscope

Most people know their birth sign according to Western astrology (even those who claim not to be interested in the subect!), but with this program you can discover not only your Chinese birth sign, but also your ascendant, your ruling element, and the sign governing the MONTH of your birth.

To run, press GOTO 0. Do not press RUN (for reasons we shall see shortly). The four inputs required are all numeric.

YEAR: e.g.1959
MONTH: e.g. 4 (do not type April!)
DATE: e.g. 23
HOUR: MUST be according to 24 hour clock. e.g. 1330 (for 1.30 pm) If you do not know the hour of your birth, enter any 4-digit number, but ignore the ascendant given.

Note: Whenever you declare a variable in a program, it takes up room both in the Program Area and in the Variable Store. This can be extravagant at the best of times, but when, as in this case, you have a string variable 84 characters long, it is going to take a big chunk out of 1K.

However, it is possible to declare variables in immediate mode, and then refer to them in a program. Unlike some more expensive computers, the ZX81 will SAVE variables thus declared along with the program itself.

The problem with this technique is that you must not press CLEAR or RUN, but must use GOTO... (line no. of start of program). Fortunately ZX-basic allows GOTO to be used with non-existent line numbers, so GOTO 0 is always adequate. If you find you have difficulty remembering when to use RUN and when to use GOTO 0, consider using the 'Auto-Run' facility dealt with in a later section.

CHINESE HOROSCOPE

FIRST ENTER IN IMMEDIATE MODE:-

```
LET A$="MONKEY←ROOSTERDOG←←←←BOAR←←←RAT
←←←←OX←←←←←TIGER←←RABBIT←DRAGON←SNAKE←←
HORSE←←SHEEP←←"

LET E$="METALWATERWOOD←FIRE←EARTH"

 5 PRINT "YEAR"
 10 INPUT Y
 15 PRINT "MONTH"
 20 INPUT M
 25 PRINT "DATE"
 30 INPUT D
 35 PRINT "HOUR"
 40 INPUT H
 45 CLS
 50 IF M=1 THEN LET Y=Y-1
 55 LET S=(Y-INT (Y/12)*12)*7+1
 60 PRINT "BTH.←SIGN:";A$(S TO S+6)
 65 LET S=INT ((Y-INT (Y/10)*10)/2)*5+1
 70 PRINT "ELEMENT←←:";E$(S TO S+4)
 75 IF D>21 THEN LET M=M+1
 80 LET M=M+5
 85 IF M>12 THEN LET M=M-12
 90 LET S=(M-1)*7+1
 95 PRINT "MONTH←←←←:";A$(S TO S+6)
 100 LET S=INT (((INT (H/100))+1)/2)+5
 105 IF S>12 THEN LET S=S-12
 110 LET S=S*7-6
 115 PRINT "ASCENDANT:";A$(S TO S+6)
```

Biorhythm

In this program, you are asked to enter your date of birth (format as for Chinese Horoscope) and the current date. The program then calculates how far you are into each of the three biorhythmic cycles: Physical, Emotional, and Intellectual.

These cycles are 23, 28, and 33 days long respectively. 'Peak' condition is 1/4 of the way through a cycle, and 'low' is 3/4 of the way through. The start/finish and mid-points of each cycle are believed to be 'danger' areas. (So don't bother attempting any complex computer programming on days when you get :Intellectual: 16.)

Notes

The program utilises the handy fact that ZX-basic gives every number a code of 28 more than the number itself. By an amazing coincidence, 28 happens to be the lowest number of days in a month, so the 'data string' (M$) is used to hold the number of days in each month as a single digit. M$(1) for instance is 3, and so CODE M$(1) is 31, the number of days in January. (M$ is only 11 characters long, as data for prior months only are required, so December is omitted.)

Line 155 adjusts for leap years, but by a strange quirk of the calandar, 1900 was not a leap year, so the program won't work if you are over 81! (2000 however WILL be a leap year, so it is valid for some considerable time yet.)

BIORHYTHM

```
5 LET M$="30323233232"
10 PRINT "D.O.B."
15 GOSUB 100
20 LET P=Q
25 PRINT "DATE TODAY"
30 GOSUB 100
35 LET P=Q-P
40 PRINT TAB 10;D;".";M;".";Y
45 PRINT AT 3,0;"PHYSICAL (23):";TAB 27;
     P-23*INT (P/23)
50 PRINT AT 5,0;"EMOTIONAL (28):";TAB 27;
     P-28*INT (P/28)
55 PRINT AT 7,0;"INTELLECTUAL (33):";
     TAB 27;P-33*INT (P/33)
90 STOP
100 PRINT "DAY"
105 INPUT D
110 PRINT "MONTH"
115 INPUT M
120 PRINT "YEAR"
125 INPUT Y
130 CLS
135 LET Q=0
140 FOR J=1 TO M-1
145 LET Q=Q+CODE M$(J)
150 NEXT J
155 IF Q>59 AND Y/4=INT (Y/4) THEN LET Q=Q+1
160 LET Q=Q+D+INT (365.25*(Y-1))
170 RETURN
```

If you prefer the more conventional graphical representation of biorhythms, this program prints out your chart for the next 3 weeks. Three inputs are required, these being the current biorhythm data as obtained from the previous program.

The horizontal axis of the graph represents days, the first '-' being today. The vertical axis is, of course, arbitrary.

The 3 graphs are superimposed on one another: if you find this confusing, try modifying the program to print them separately.

BIOCHART

```
5 FOR J=0 TO 23
10 PRINT AT 6,J;"-"
15 NEXT J
20 LET P=23
25 PRINT AT 0,0;"PHYSICAL"
30 GOSUB 100
35 LET P=28
40 PRINT AT 0,0;"EMOTIONAL"
45 GOSUB 100
50 LET P=33
55 PRINT AT 0,0;"INTELLECTUAL"
60 GOSUB 100
90 STOP
100 INPUT N
105 FOR J=0 TO 46
110 PLOT J,SIN ((J+(2*N))*PI/P)*10+30
115 NEXT J
120 RETURN
```

Decisions

Finally a less serious method of fortune telling! The inspiration for this program came from one of those swinging pendulum executive toys, which just goes to show that you can get programming ideas from all sorts of sources!

Notes

Line 5: The program doesn't really care what question is asked, so dimensioning Q$ to 1 element allows the user to type away to his heart's content, without consuming valuable memory. This technique can be used in many more serious applications, where the action the computer takes is dependant only on the first letters of the user's input.

Line 135: The letter Z is not a misprint! True, the variable Z has not occured elsewhere in the program, so line 135 will cause the program to terminate with error code 2/135 (undefined variable), but of course this is exactly what we want. (If you feel uneasy about deliberately inducing errors in your programs, note that the Sinclair manual refers to 'Report' Codes, not Error Codes!)

Lines 200-205: A dummy loop, which we prefer to PAUSE because of the awful screen flicker. It's more expensive on memory, of course, but in this case that is counteracted by the loop-control variable ending up with a value which is useful elsewhere. At the end of the loop, X=9 (not 10 !!), which is precisely the value required in line 35, so that option 10 is not chosen twice. Don't waste loop-control variables!

DECISIONS

```
5 DIM Q$(1)
10 SCROLL
15 PRINT "ASK ANY QUESTION"
20 INPUT Q$
25 IF Q$="←" THEN GOTO 20
30 LET X=10
35 CLS
40 LET X=INT (RND*X)+1
45 IF X=1 THEN PRINT "YES"
50 IF X=2 THEN PRINT "NO"
55 IF X=3 THEN PRINT "MAYBE"
60 IF X=4 THEN PRINT "HOW SHOULD I KNOW?"
65 IF X=5 THEN PRINT "NEVER"
70 IF X=6 THEN PRINT "OF COURSE"
75 IF X=7 THEN PRINT "THAT IS UP TO YOU"
80 IF X=8 THEN PRINT "POSSIBLY"
85 IF X=9 THEN PRINT "I GIVE UP-WHAT DO
          YOU THINK?"
90 PRINT
95 IF X=10 THEN GOTO 200
100 PRINT "ARE YOU SATISFIED WITH MY
          REPLY?"
105 INPUT Q$
110 IF Q$="N" THEN GOTO 30
115 CLS
120 PRINT "ANY MORE?"
125 INPUT Q$
130 IF Q$="Y" THEN RUN
135 PRINT "O.K.BYE";Z
200 PRINT "THIS IS TRICKY...LET ME
          THINK..."
205 FOR X=100 TO 10 STEP-1
210 NEXT X
215 GOTO 35
```

Printing with More Frills

(If you don't know where the 'more' comes from, you haven't read the Sinclair manual properly!)

Three routines to enhance your screen displays.

Ad.

So named because we sometimes leave this running in our shop window. You can, of course, change lines 15, 30, 45, and 55 to anything you like.

Notes

The main routine (lines 1000-1070) prints the strings OUTWARDS FROM THE CENTRE!!

The secondary routine (lines 2000-2030) inverts the strings on alternate printings.

K simply alternates between 'true' and 'false' (line 930) to call the inversion routine when required.

N is the horizontal print coordinate, and can be varied at will.

Always ensure that A$ contains an even number of characters, or the first one will be lost. If you like, you could add a routine to add a space to A$ whenever necessary, but it is far more economical to do that manually.

AD.

```
5 SLOW
10 LET K=0
15 LET A$="SINCLAIR ZX 81"
20 LET N=2
25 GOSUB 1000
30 LET A$="BY SINCLAIR RESEARCH"
35 GOSUB 1000
40 LET N=N+5
45 LET A$="SOFTWARE FROM:"
50 GOSUB 1000
55 LET A$="V+H←←182C KINGSTON RD. STAINES"
60 GOSUB 1000
900 FOR I=1 TO 100
910 NEXT I
920 CLS
925 REM :DELETE 920 FOR OTHER EFFECT
930 LET K=NOT K
940 GOTO 15
1000 LET L=LEN A$
1010 LET N=N+2
1020 IF K THEN GOSUB 2000
1030 FOR I=1 TO L/2
1040 PRINT AT N,16-I;A$(L/2+1-I)
1050 PRINT AT N,15+I;A$(L/2+I)
1060 NEXT I
1070 RETURN
2000 FOR J=1 TO L
2010 LET A$(J)=CHR$ (CODE (A$(J))+128)
2020 NEXT J
2030 RETURN
```

If you are unfamiliar with machine code listings, the procedure is as follows.

1: Spot the program! The only part you have to enter is the second column beginning 06 16 2A... and ending 10 EF C9. It doesn't matter that whether there are 1, 2 or more 'bytes' per line: they are arranged like this to help you understand the program, which, with the help of the Sinclair manual, you should be able to do.
2: Count the number of bytes; in this case there are 23.
3: Enter a Machine Code Loader. There is a very simple one listed later in the book if you don't have one already. Line 1 should contain 23 (for this routine) X's, and LIMIT in line 10 should be set to 16536.
4: Run the loader, and enter the 23 two-character bytes.
5: (optional) Poke the routine to line 0 (using another routine you'll find later!) and delete the loader.
6: Finally, if you are wondering if all this was worth the effort, enter the DEMO program which follows.

Call the routine by: LET K=USR 16514.

INVERSE PRINT

```
00    06 16      START :LD B,22          ;22 LINES
02    2A 0C 40          LD HL,(16396);D.FILE
05    23         LOOP  :INC HL           ;NEXT POSN.
06    7E                LD A,(HL)       ;CHR. CODE
07    FE 76             CP 118          ;IS IT 'N/L'
09    28 09             JRZ 9           ;IF SO,SKIPNL
0B    FE 00             CP 0            ;IS IT SPACE
0D    28 03             JRZ 3           ;IF SO,SKIPSP
0F    C6 80             ADD A,128       ;INVERT CHR.
11    77                LD(HL),A        ;PRINT
12    18 F1      SKIPSP:JR 241          ;NEXT CHR.
14    10 EF      SKIPNL:DJNZ 239        ;NEXT LINE
16    C9                RET             ;RETURN
```

Assuming that you have entered the machine code Inverse Print program, this demonstration shows the fantastic difference in speed between machine code and Basic.

Lines 100-140 are the Inverse Print routine translated into Basic. (Not quite: the m/c routine searches the whole screen; this Basic version just covers the screen area which we know is in use, but the principle is exactly the same.)

Because the m/c routine changes the whole screen INSTANTLY, option 2 introduces a delay loop (lines 50-55) to allow time for you to take your finger off the key. Option 3 omits this loop, so holding down key 3 creates the flashing effect.

DEMO

```
5 REM -M/C INVERSE PRINT-
10 PRINT AT 1,1;"INVERSE PRINTING DEMONSTRATION"
15 PRINT AT 4,5;"1: BASIC"
20 PRINT AT 6,5;"2: MACHINE CODE"
25 PRINT AT 8,5;"3: ←←←/←←←←/←←FLASHING"
30 IF INKEY$<"1" OR INKEY$>"3" THEN GOTO 30
35 IF INKEY$="1" THEN GOTO 100
40 LET K=USR 16514
45 IF INKEY$="3" THEN GOTO 30
50 FOR K=1 TO 30
55 NEXT K
60 GOTO 30
100 LET H=PEEK 16396+256*PEEK 16397
110 FOR K=H TO H+102
120 IF PEEK K=0 OR PEEK K=118 THEN GOTO 140
130 POKE K,PEEK K-128
140 NEXT K
150 GOTO 30
```

Data Print

There have been many programs published (and you have probably written one yourself) which enable you to 'draw' on the screen. But how can you SAVE your masterpiece? The idea of Data Print is to store your graphic displays in a data string, and the program also offers a couple of useful aids to simplify your drawing.

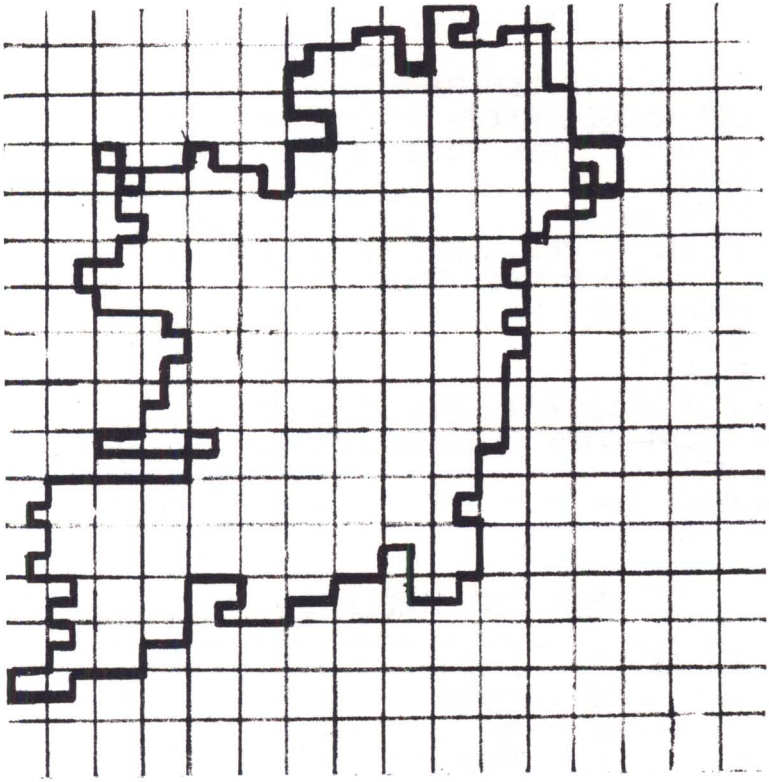
If you want a number of blank spaces, just enter into the string that number. Similarly, if you want a number of black spaces, enter the INVERSE number. To force a line feed, i.e. if all the remainder of the line is blank, enter N.

Before you run the program, you must first create your drawing. Use a sheet of graph paper or the ZX Graphics Pad (published by V&H). (If you are a real cheapskate you can photocopy page 123 of the Sinclair manual, but don't tell Clive we suggested that, because it's copyright!).Then enter the data string, using the facilities mentioned above for large areas of black or white. You will probably find it easier to 'read' direct from your drawing, rather than write down the string first, but don't forget the N at the end of each line.

The example we have chosen to illustrate this very useful technique is a pretty appalling map of Ireland! None of us at V&H are artists, and this is the best we could come up with! If any of you can do any better, let us know and we will use the best example(s) in the next edition of this book.

N.B. If your data string is not much longer than 'Ireland', it will fit in the program as line 10: otherwise enter it in immediate mode and use GOTO 0 to run.

Please note that, for the Ireland map, we have abandoned the usual listing conventions. The first number should be entered as a normal number. The characters in the central column are either inverse numbers (when underlined) or otherwise graphics. The N of course is a normal N. The entire string should be entered continuously with NO SPACES.



Think you could do any better?

(Could you possibly do any worse?)

DATA PRINT

```
20 FOR J=1 TO LEN A$
30 LET N=CODE A$(J)
40 IF N<>51 THEN GOTO 70
50 PRINT
60 GOTO 90
70 IF (N>28 AND N<38) OR (N>156
        AND N<166) THEN GOTO 100
80 PRINT CHR$ N;
90 NEXT J
99 STOP
100 FOR K=128*(N>128) TO N-29
110 PRINT CHR$ (128*(N>128));
120 NEXT K
130 GOTO 90
```

DATA FOR 'IRELAND'

```
10 LET A$="7    34E34       N
           6    Q1W25       N
           6    75          N
           2    Y6W46R      N
           2    28E1        N
           1    3Q7E        N
           2    7R6E        N
           3    36E         N
           3    Q65         N
           2    77Q51       N
                38E         N
                37R1        N
           1    Q2317
           1    21          N
           1    Q17         N
                71"
```

Casino Gambling

Now we know that many ZX81 owners are still at school, and we are certainly NOT trying to encourage gambling. On the contrary, you will lose so much hypothetical money playing the games in this section, and the ZX81 will lose so much playing roulette, that you should be convinced that the only winners are the casino owners! Nevertheless, casino games always have a fascination for computer users, and so we are including a few of the more unusual examples for you to play. First, though, let's give the ZX81 a night out, and see if it can break the bank at...

Roulette

Many people claim to have created winning systems by using a computer. Can it be done? That is up to you to discover; what we are going to do is provide a 'standard' roulette program, and then use it to test 4 of the most popular systems. Then you can slot you own ideas in, and see if you can come up with a winner.

N.B. The ZX81 is no mug, and it is not going to try for spectacular wins by playing the 'long shots'. It knows that the best odds in roulette are for the 'even money' chances: Pair, Impair, Rouge, Noir, Manque and Passe (or, in English, Even, Odd, Red, Black, 1-18, 19-36). It doesn't matter which of these is chosen, so for this series of games, the computer will always bet 'Even'.

The Standard Program

This is a 'non-system' in which the computer bets a constant stake throughout. It should show a small but ever worsening loss. (Problem: What is the expected loss over 370 games for a constant $1 stake?)

Notes

N is the actual winning number, which is not required unless it happens to be 0, in which case half of the stake is returned (lines 150 & 270).

X is the winning number, expressed as even (X=1) or odd (X=0). X is also set to 0 if N=0, so that, for the systems, 0 is considered to be a losing spin.

The 'menu' (lines 50-70) is there in case you

don't trust the computer to play fairly! Select option 2 to input numbers from, say, a toy roulette wheel (or recorded from an actual casino, if you want to get really fanatical about this!).

Line 130 is just an input check, and serves little purpose. (None whatever if you select option 1.) Similarly, line 210 is purely cosmetic. Both these lines may be deleted, as may lines 50, 60, 70 and 120, if you require more room to test a particular system.

SAVE a copy of the standard program, as it is the basis for all of the next 4.

Martingale

This system involves doubling the stake after a loss, and reducing it to 1 after a win. It is infallible! (But see the comments on page 42.)

Reverse Martingale

This is the exact reverse of the above, and it should make a small but progressive loss. However, there will be the occasional big win, which SHOULD (!) leave you in profit. The system fared badly in our tests - possibly because it's most effective use is not over a set number of games, but just up to a point when the user is in profit. As this seems to go against the psychology of gambling, the authors could see little value in the system.

D.Alambert

Raise the stake by 1 unit after a loss. Reduce it by 1 after a win. This tends to keep the stakes lower than with the Martingale, but notice that it did produce the most spectacular losses in the series of test games!

Labouchere

This was so amazingly successful that I think I'll head straight to the nearest casino, rather than explain to you how it works!

(And while you are working that out, see if you can see the flaw in all these systems, and why you can't actually break the bank using any of them.)

ROULETTE

```
10 RAND
20 LET P=0
30 LET T=1
40 LET S=T
50 PRINT "1:COMP WHEEL"
60 PRINT "2:YOUR←←←"
70 INPUT A
80 CLS
90 PRINT "I BET EVENS:$";S
100 LET P=P-S
110 LET N=INT (RND*37)
120 IF A=2 THEN INPUT N
130 IF N<0 OR N>36 THEN GOTO 120
140 LET X=0
150 IF NOT N THEN GOTO 270
160 IF N/2=INT (N/2) THEN LET X=1
170 PRINT N
180 PRINT "I←";
190 GOSUB 400-50*X
200 PRINT "NET $";
210 IF SGN P=1 THEN PRINT "+";
220 PRINT P,,"AFTER←";T;"←SPINS"
230 FOR X=1 TO 50
240 NEXT X
250 LET T=T+1
260 GOTO 75
270 LET P=P+S/2
280 GOTO 170
350 LET P=P+S+S
360 PRINT "WIN"
370 RETURN
400 PRINT "LOSE"
410 RETURN
```

MARTINGALE

```
365 LET S=1
405 LET S=S*2
```

REVERSE MARTINGALE

```
365 LET S=S*2
366 IF S>256 THEN LET S=1
405 LET S=1
```

D.ALAMBERT

```
365 LET S=S-1
366 IF S=0 THEN LET S=1
405 LET S=S+1
```

LABOUCHERE

```
40 REM ** DELETE LINE 40 **
130 REM ** DELETE LINE 130 **
73 LET A$="↑1274↑"
75 LET S= CODE A$(1)+CODE A$(LEN A$)
255 IF A$="" THEN GOTO 73
365 LET A$=A$(2 TO LEN A$-1)
403 IF S>255 THEN LET S=255
405 LET A$=A$+CHR$ S
```

RESULTS ANALYSIS

All 5 roulette programs were tested over a series of 8 'sessions', each session consisting of 370 games. The results were as follows:

	1	2	3	4	5	6	7	8	AV.
Std.	-69	-54	-17	-5	8	16	-22	-36	-22
Mal.	212	262	207	203	186	223	178	210	210
R.M.	-167	-178	-177	90	-164	92	91	88	-41
D.A.	-264	-655	48	184	-673	212	80	78	-124
Lab.	735	664	766	603	773	512	744	227	628

Conclusions:

As only one of the 'control' sessions came close to the expected loss, our main conclusion was that even 2960 games was insufficient for serious statistical analysis!

However, we did conclude that, provided you go into a casino with the understanding that you will almost certainly lose money, then both the D.Alambert and Labouchere would probably provide a more entertaining evening than simply random betting.

Before you get too enthusiastic about LABOUCHERE, we must tell you that in one final test session, this produced a loss of $3205 !!!

For futher details, see page 42.

Boule

If all that theorising hasn't put you off gambling for life, here is a game which might! Boule is a sort of 'mini-roulette' which is often found in France, but apparently illegal in England because the casino advantage is so high.

There are 9 numbers on the wheel; 1, 3, 6, and 8 are black; 2, 4, 7, and 9 are red, or in the ZX81 version, white. 5 has a function similar to 0 in roulette. i.e. It can be backed as a number but not in any of the combination bets. Unlike roulette, however, all bets except those on the number 5 lose completely whenever 5 is spun. The bets available are:-

Noir (any of the 4 black nos.)
Blanc (any of the 4 white numbers)
Manque (any of the low numbers 1-4)
Passe (any of the high numbers 6-9)
Pair (any of the even numbers)
Impair (any of the odd numbers EXCEPT 5)
(All the above pay even money)
Any number individually (inc. 5): pays 7 to 1

There is a board for playing Boule printed on page 38.

Notes

The main program loop (lines 15-65):
a) Prints the numbers of the 'wheel' (lines 30 & 45)
b) Is of variable duration (line 20)
c) Slows down towards the end (lines 50 & 55)
d) Returns the winning number in N

Line 60 is another form of pause, again preferable to the PAUSE statement. It can be omitted if you want to play the game at a faster pace.

The remainder of the program prints the winning bets. Line 160 is included before the re-run option, as otherwise there may not always be room for even a null input.

BOULE

```
5 RAND
10 LET N=0
15 LET A$="1234*6789"
20 LET M=(18*(INT (RND*5)+1))+(INT
        (RND*9)+1)
25 FOR J=1 TO M
30 PRINT AT 0,N*3;"←"
35 LET N=N+1
40 IF N>9 THEN LET N=N-9
45 PRINT AT 0,N*3;A$(N)
50 FOR K=M-10 TO J
55 NEXT K
60 LET D=RND*RND
65 NEXT J
100 PRINT N;" PAYS 7-1"
105 IF N=5 THEN GOTO 155
110 LET A$="NOIR"
115 IF N=2 OR N=4 OR N=7 OR N=9
        THEN LET A$="BLANC"
120 GOSUB 200
125 LET A$="MANQUE"
130 IF N>5 THEN LET A$="PASSE"
135 GOSUB 200
140 LET A$="PAIR"
145 IF N/2<>INT (N/2) THEN LET A$=
        "IM"+A$
150 GOSUB 200
155 PRINT "ALL OTHER BETS LOSE"
160 CLEAR
165 INPUT A$
170 CLS
175 RUN
200 PRINT A$;" PAYS EVEN"
210 RETURN
```

Noir et Blanc

This is another obscure French game, this time normally played with cards. The dealer (ZX81) lays a horizontal row of cards, and continues adding to the row until the 'pip-count' totals more than 30 (A=1; picture cards=10). He then deals a second row, again until it totals more than 30. The winning row is the row with the LOWEST total. The top row is called Noir, and the second row is called Blanc.

Four bets are allowed, which all pay even money:

Noir :The top (Noir) row will win
Blanc: The second (Blanc) row will win
Couleur: The colour of the first card dealt will be the same as the NAME of the winning row
Inverse: The colour of the first card dealt will NOT be the same as the name of the winning row

It is these last two bets which give the game an appeal totally out of proportion with it's complexity!

N.B. If both rows total the same, it is a draw and all bets are returned, UNLESS they both total 31, in which case there is a REFAIT. i.e. All bets remain on the table, to be returned only if they win on the next deal.

Notes

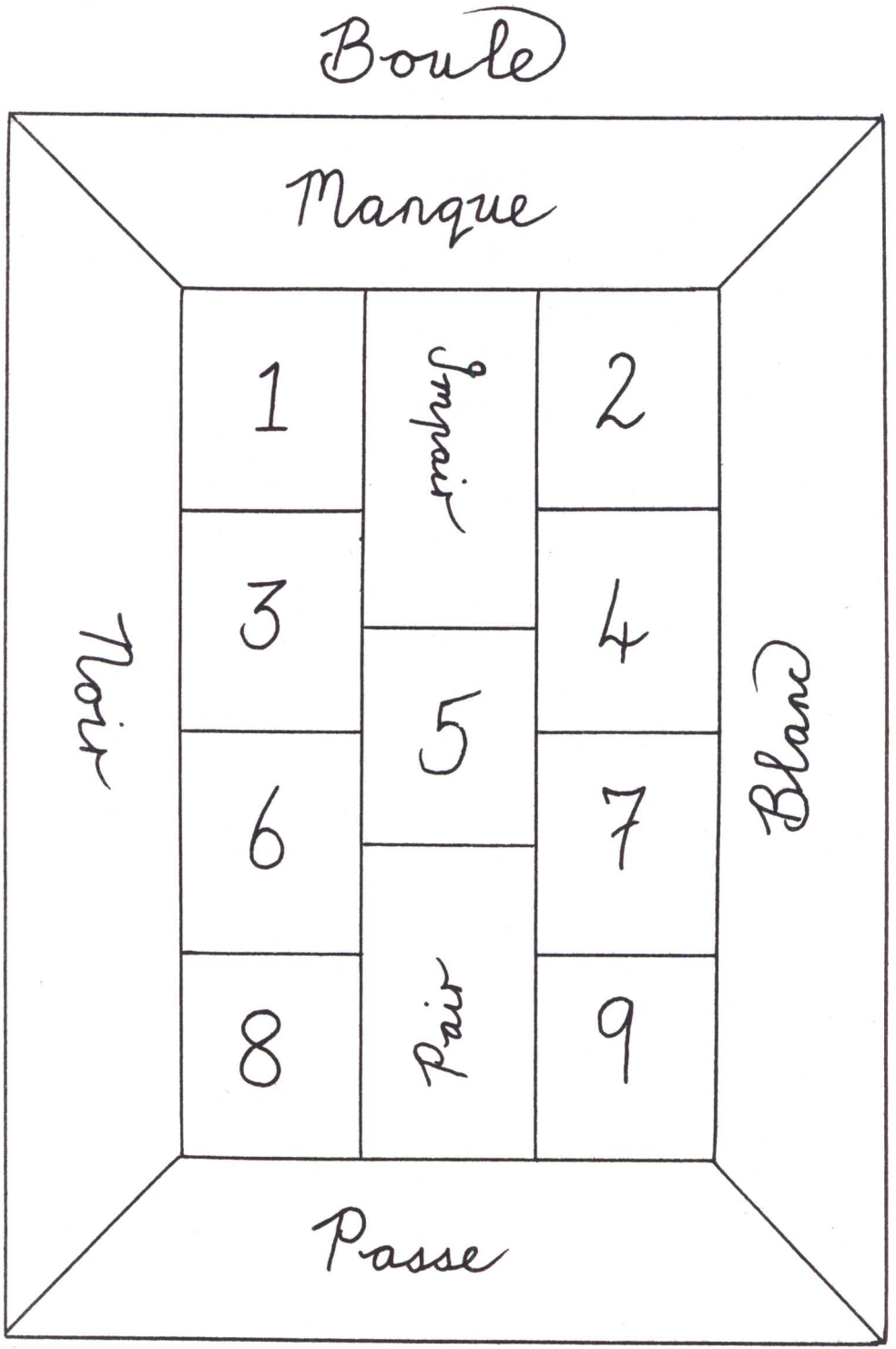
Again there is a main program loop (lines 25-90) and a 'tail' which prints out the winners. Line 10 is another form of data statement; rather than hold the cards in a string, putting them in the first program line makes them easily accessible by PEEK (line 45).

Line 50 randomly inverts the selected card, so that the entire 'pack' need only contain 13 cards.

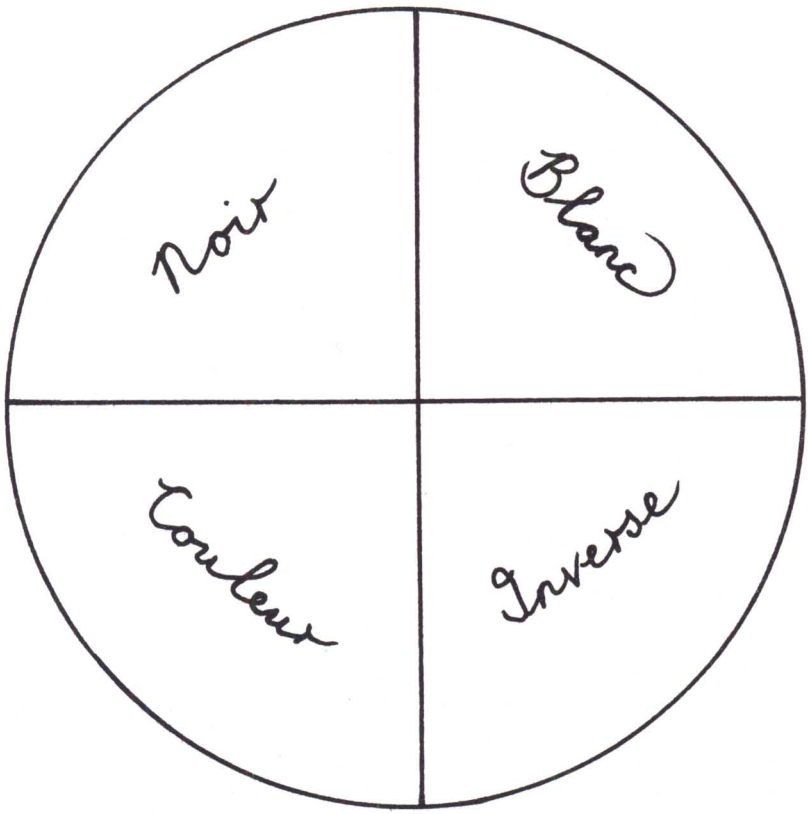
The winners of the two pairs of bets are determined by the Boolean logic of lines 105 & 120. The couleur/inverse bet is settled in line 120 by first PEEKing at the first card (6 positions into the display file, because the display always starts with N, O, I, R and Newline), and setting a flag (X=1) if the card is black. (Otherwise X=0). The flag J has already been set to 1 if the Noir line wins (line 105), so the Couleur bet will win if X and J are the same, regardless of whether they are both 0 or 1.

NOIR ET BLANC

```
10 REM A23456789TJQK
15 DIM A(2)
20 PRINT "NOIR"
25 FOR J=1 TO 2
30 LET X=INT (RND*13)+1
35 LET Y=X
40 IF Y>10 THEN LET Y=10
45 LET A$=CHR$ (PEEK (16513+X))
50 IF RND<.5 THEN LET A$=CHR$ (CODE
       A$+128)
55 PRINT A$;"←";
60 LET A(J)=A(J)+Y
65 IF A(J)<31 THEN GOTO 30
70 LET A(J)=A(J)-30
75 PRINT TAB 20;"= ";A(J)
80 LET A$="BLANC"
85 IF J=1 THEN PRINT A$
90 NEXT J
100 IF A(1)=A(2) THEN GOTO 160
105 LET J=A(1)<A(2)
110 IF J THEN LET A$="NOIR"
115 PRINT A$
120 LET X=PEEK ((PEEK 16396+256*PEEK
       16397)+6)>128
125 LET A$="INVERSE"
130 IF X=J THEN LET A$="COULEUR"
135 PRINT A$
140 INPUT A$
145 CLS
150 RUN
160 LET A$="DRAW"
165 IF A(1)=1 THEN LET A$="REFAIT"
170 PRINT A$
175 GOTO 140
```


Boule
Manque
1
impair
2
3
4
5
Noir
Blanc
6
7
8
pair
9
Passe

Noir et Blanc



Craps

We must state at once that this is the name of the game, and not a description of it. (Though some readers might disagree!)

Craps is a typical example of how a simple activity like rolling a couple of dice, can be turned into a game so amazingly complex that you would need an IBM Megacomputer to hold all the possible options and variations. Faced with trying to cram such a game into 1K, there are two possible approaches, and indecisive as ever, we have adopted both of them!

In DICE, the computer merely handles the mechanics of the game (i.e. just rolls the dice); CRAP SHOOTER eliminates all players except one, thereby reducing the variety of bets considerably, as there are no 'side bets' and, of course, the 'shooter' is not allowed to bet that he will lose.

Dice

Dice throwing programs for the ZX81 are not uncommon, but we have included this one because it has the advantage of allowing the number of dice to be thrown at once to be pre-selected. Up to 4 dice will be shown on the screen at one time. If a larger number has been requested, then the screen will clear after the pause loop (lines 164-166) and the next 4 will be printed. At the end of each 'throw', simply press NEWLINE to throw again.

The line numbers of this program have not been 'tidied up', so you can see where the pre-selection routine has been added to the original program. If you merely require, say, two dice, as you would to play Craps, you can delete all the 'late additions' (all line numbers which don't end in 0) and change D in line 180 to the required number.

DICE

```
4 PRINT "NO. OF DICE?"
6 INPUT D
10 RAND
20 DIM A$(6,15)
30 LET A$(1)="+++++++▒"
40 LET A$(2)="▒+++++++++++++▒"
50 LET A$(3)="▒+++++++▒+++++++▒"
60 LET A$(4)="▒+++▒+++++++▒+++▒"
70 LET A$(5)="▒+++▒+++▒+++▒+++▒"
80 LET A$(6)="▒+++▒▒▒+++▒▒▒+++▒"
90 CLS
100 LET N=0
110 LET X=INT (RND*6)+1
120 FOR J=1 TO 11 STEP 5
130 PRINT TAB N;A$(X,J TO J+4)
140 PRINT
150 NEXT J
160 LET N=N+8
162 IF N/32<>INT (N/32) THEN GOTO 170
164 FOR X=0 TO N
166 NEXT X
168 CLS
170 PRINT AT 0,0;
180 IF N<8*D THEN GOTO 110
190 INPUT 0$
200 GOTO 90
```

Crap Shooter

As already stated, this is a one-player version of craps, with no side bets. The shooter (you) starts with $100 and stakes $1 per game. A 'game' proceeds as follows:

1st Roll:

7 or 11: NATURALS: Win immediately
2,3 or 12: CRAPS: Lose immediately
Anything else: COME OUT ROLL: Proceed to roll #2

Second and subsequent rolls:

Continue rolling until you roll either:-
Your come out roll (again): WIN
7: LOSE

Note that 7, the most common roll with two dice, wins if rolled first, but otherwise loses. 2,3,11 and 12 have no significance other than for the first roll.

Final thoughts on Roulette

Before we leave gambling completely, here are the answers to the questions posed on page 29 & 30.

The expected loss over 370 games (constant $1 stake) is $5.

The reason that none of the systems work, is that all casinos have a maximum AND a minimum stake. i.e. Even if you start at the minimum stake, any system which involves increasing the stake is eventually going to exceed the maximum, at which point, of course, it will break down.

The Labouchere tends to avoid the problem of reaching the maximum by reducing the stake more often than increasing it. (Almost twice as often - if you still can't see how it works from the listing, drop us a line and we will explain it.) As you can see from page 33, it works quite often...but when it fails, it really fails!!

CRAP SHOOTER

```
10 PRINT TAB 10;"CRAP SHOOTER"
15 LET N=101
20 LET C=0
25 LET N=N-1
30 PRINT AT 15,0;"YOU HAVE $";N
35 LET Y=0
40 PRINT
45 PRINT "PRESS N/L TO ROLL"
50 INPUT Q$
55 CLS
60 LET X=INT (RND*6)+1
65 PRINT X,
70 LET Y=Y+X
75 IF Y=X THEN GOTO 60
80 PRINT
85 IF C THEN GOTO 105
90 IF Y<4 OR Y=12 THEN GOTO 200
95 IF Y=7 OR Y=11 THEN GOTO 250
100 LET P=Y
105 LET C=C+1
110 PRINT
115 PRINT "COME OUT ROLL: ";P
120 IF C=1 THEN GOTO 30
125 PRINT
130 IF Y<>P THEN GOTO 160
135 PRINT
140 PRINT TAB 10;"**POINT MADE**"
145 LET N=N+2
150 GOTO 20
160 IF Y=7 THEN GOTO 200
165 PRINT "ROLL: ";Y
170 GOTO 30
200 PRINT
205 PRINT TAB 13;Y;" CRAP"
210 GOTO 20
250 PRINT
255 PRINT TAB 10;"**NATURAL** ";Y
260 GOTO 145
```

Data Files

One feature noticeably absent from the standard ZX81 is the ability to read and write external data files. However, by extending an idea we have used previously (that of SAVEing variables along with a program), we can effectively emulate a data file by SAVEing a copy of the program, with the LATEST value of it's variables, AFTER it has been run.

The next two programs are for fun. Later, in the Business section, we will put this principle to more serious use.

Top Ten (Chart Data)

This program reflects the author's joint obsession with pop charts and statistics (?!) Every year, the pop music papers publish a 'points table' of artists who have had hit records that year. Each week they score 30 points for a record at no.1, 29 for no.2 etc. Each year, I am so fascinated with these 'chart of charts', that I vow not to wait another 12 months for the next one, but to keep my own weekly up-date of the points table. Even with the aid of a computer, this is a formidable task, and rather beyond the ability of 1K. This program, therefore, is a 'mini' points table; you select ten of your favourite artists, and follow their relative chart progress throught the year.

Notes

The program has been squeezed considerably to allow it to hold data on ten artists. Line 6 is a long winded but, believe it or not, more economical way of saying LET C=10. (Check this with the 'Byte-Counting' routine on page 65) Similarly, line 7 sets D=1. The first six lines have been given these low line numbers just to allow line 115 to say GOTO C instead of GOTO 10!

G$ has been dimensioned to hold the first 10 characters (only) of the 10 artist's names. (line 8)

Lines 5, 10, and 25 allow entry of artist's names on the first run only. On subsequent runs, A$ is a null string (as execution commences at line 10), so line 25 is skipped.

TOP TEN (CHART DATA)

```
5 LET A$="NAME/"
6 LET C=CODE "↑S↑"
7 LET D=C/C
8 DIM G$(C,C)
9 DIM S(C)
10 PRINT AT C,D;" INPUT ";A$;"POSN."
15 FOR N=D TO C
20 SCROLL
25 IF A$<>"" THEN INPUT G$(N)
30 PRINT G$(N),
35 INPUT X
40 IF X THEN LET S(N)=31-X+S(N)
45 PRINT S(N)
50 NEXT N
55 FOR N=D TO C
60 SCROLL
65 NEXT N
70 PRINT "PRESS S TO SAVE DATA"
75 LET A$=""
80 IF INKEY$<>"S" THEN GOTO 80
85 SAVE "TT"
90 CLS
95 PRINT
100 FOR N=D TO C
105 PRINT G$(N),S(N)
110 NEXT N
115 GOTO C
```

Computer Dating

The ZX81 can do many things; whether it can help create perfect marriages remains to be seen! This is intended as a 'fun' program, to be used among your school or work mates or at parties. But if any successful partnerships result, we'd be very pleased to hear! (For say parties, delete lines 500 & 505 - the ZX81 is nothing if not versatile!)

Preliminaries

At the back of this book, you will find a questionnaire. Photocopy and distribute as many copies as possible.

Complete the 'input document' at the foot of each questionnaire. Box 1 should always contain M or F. Boxes 2-7 should contain 1 or 2, according to the item chosen. Boxes 8-10 should contain the 3 'likes' letters (in alphabetical order); boxes 11-13 the 3 'dislikes'.

Establish a data-bank as follows:-
Run the program, selecting menu item 1, and enter the relevent data. (Name/Box 1/Boxes 2-7/Boxes 8-13) Repeat this until you have SAVEd data on all the applicants. Note that this part of the program will end by naming each applicant as his/her ideal partner (fair enough!). Enter NEWLINE to proceed to the next applicant.

Now rewind the tape, and you are ready to start matchmaking. Select menu item 2, and enter the first applicant's data. The program will search through the tape for suitable partners. (If it fails to find a match for everyone, it serves them right for being thick, humourless disco-going poetry fanatics!)

Notes

The interesting feature of this program is that it uses the program NAME as a data store, a concept which you should be able to use elsewhere.

Lines 500-505 reverse the first element of the data string P$ (1 is female: 0 is male) so that a match of the opposite sex is found.

COMPUTER DATING

```
5 PRINT TAB 9;"COMPUTER DATING"
10 PRINT "1: DATA"
15 PRINT "2: PARTNERS"
20 INPUT M
25 PRINT "NAME"
30 INPUT N$
35 PRINT "SEX M/F"
40 INPUT S$
45 LET P$="1"
50 IF S$="M" THEN LET P$="0"
55 PRINT "PERSONALITY CODE "
60 GOSUB 200
65 PRINT "LIKE/DISLIKE CODE "
70 GOSUB 200
75 IF M=2 THEN GOTO 500
80 PRINT "START TAPE: PRESS S TO SAVE DATA"
85 IF INKEY$<>"S" THEN GOTO 85
90 SAVE P$
95 CLS
100 PRINT "YOUR IDEAL PARTNER IS"
105 PRINT N$
110 PRINT "WANT AN ALTERNATIVE?"
115 INPUT N$
120 CLS
125 IF N$<>"Y" THEN RUN
130 GOTO 510
200 INPUT S$
210 IF LEN S$<>6 THEN GOTO 200
220 LET P$=P$+S$
230 RETURN
500 LET M=NOT VAL P$(1)
505 LET P$(1)=STR$(M)
510 PRINT "PRESS L TO LOAD DATA"
515 IF INKEY$<>"L" THEN GOTO 515
520 LOAD P$
```

Check Digit

This program has nothing whatever to do with data files, and appears in this section due to a computer error. (Well, that's the excuse everyone else uses!)

A check digit is a very important but much neglected form of data validification. The LEAST serious consequences of invalid data being entered into a program are that the program will crash or will produce inaccurate results (the well known GIGO phenomenon: Garbage In, Garbage Out). The consequences can be far more serious if the invalid entry is made fraudulently by an unauthorised person. The principle of a check digit is that, whenever a numeric entry is required (typically a stock code number, for example), the final digit of the number is in some way related to the rest. Therefore a randomly chosen entry will not be accepted, as it does not satisfy the check digit's validification rules.

The simplest form of check-digiting is to add the digits of a number together (reducing if necessary to a single digit, as we did in 'Numerology'), and append the result to the original number (thus 153 becomes 1539). This program adopts a more complex approach, and should appeal to the more security-conscious readers.

First you invent a 'code': any series of numbers, as long or as short as you like. Then enter your number to be 'check-digitised'. The check digit is formed by taking each digit of the number in turn, multiplying it by the corresponding digit in the code, and then adding these numbers together as before.

Notes

The routine at lines 45-60 temporarily extends the check code in cases where the number entered is longer than the code.

To run the program for a series of numbers, press newline after each 'security number' appears. To change the check code, enter N.

CHECK DIGIT

```
5 LET A$=""
10 PRINT "CHECK CODE ",
15 IF A$="" THEN INPUT A$
20 PRINT A$
25 PRINT "NUMBER",
30 INPUT B$
35 PRINT B$
40 LET Y=0
45 LET C$=A$
50 IF LEN C$>=LEN B$ THEN GOTO 65
55 LET C$=C$+"1"
60 GOTO 50
65 FOR J=1 TO LEN B$
70 LET X=VAL C$(J)*VAL B$(J)
75 GOSUB 150
80 LET Y=Y+X
85 NEXT J
90 LET X=Y
95 GOSUB 150
100 PRINT "SECURITY CODE ",B$;X
105 PRINT AT 10,0;"N/L= SAME CODE "
110 PRINT "  N= NEW CODE "
115 PRINT "  S= STOP "
120 INPUT Q$
125 CLS
130 IF Q$="N" THEN RUN
135 IF Q$="S" THEN STOP
140 GOTO 10
150 IF X<=9 THEN RETURN
155 LET X=X-9
160 GOTO 150
```

Business Programs

We are not suggesting that the 1K ZX81 is the BEST computer for business use, but neither have we time for suggestions that it is 'just a toy'.

The first program in this section is of the 'extended calculator' variety: you CAN work out you VAT with a calculator, but it's a lot easier on the ZX81. BANK REC. is more 'personal finance' than 'business': again you could do this with a calculator, but many people don't bother. The ZX81 not only makes it easier, but also more fun. CREDIT CARDS is a very useful program to assist the retailer, for whom credit card sales often cause more problems than they are worth. Finally, the two programs PAYROLL and PAYSLIP should convince you, if you still have any doubts, what a fantastically powerful tool you have bought for your £70!

V.A.T.

No explanations necessary. The most important feature of this program is that it is menu-driven, so that you can calculate VAT either from gross OR net figures.

Line 5 is the current VAT rate, and is the only one you should ever have to change.

Line 500 is one you should memorise and use everywhere. It rounds floating point figures to the nearest hundredth (i.e. £.p)

V.A.T.

```
5 LET V=.15
10 LET V=V+1
15 PRINT AT 3,10;"VAT CALCULATOR"
20 PRINT AT 7,0;"1: VAT FROM GROSS"
25 PRINT AT 9,0;"2: VAT FROM NET"
30 LET I$=INKEY$
35 IF I$<"1" OR I$>"2" THEN GOTO 30
40 CLS
45 GOTO 100*VAL I$
100 PRINT "GROSS"
110 INPUT G
120 LET T=G/V
130 GOSUB 500
140 LET N=T
150 GOTO 250
200 PRINT "NET"
210 INPUT N
220 LET T=N*V
230 GOSUB 500
240 LET G=T
250 CLS
260 PRINT AT 3,5;"GROSS",G
270 PRINT AT 5,5;"NET",N
280 PRINT AT 7,5;"VAT",G-N
290 PRINT AT 20,4;"PRESS NEWLINE TO CONTINUE"
300 INPUT I$
310 CLS
320 RUN
500 LET T=INT (T*100+.5)/100
510 RETURN
```

Bank Rec.

A useful program for those of you who don't like paying for bank statements too often. When you DO get a bank statement, look out for bank charges and any other items you haven't included, and make the appropriate adjustment on you next run. You have one opportunity to put in an adjustment figure, at the 'Standing Order ETC.' entry.

Notes

Line 220 prevents running out of memory by having too big a display file.

Line 260 may look confusing because of all those A's, but all it does is leave your entry positive when the first letter of A$ is A (i.e. A$="AMOUNT"). Otherwise the entry is taken as negative.

The program uses the Auto-run feature (actually 'auto-goto 30'!) so after you have SAVEd your data, the words 'NUMBER OF CHEQUES' appears. Enter an alpha character to break. If at any time you want to set a new opening balance, break immediately after loading, and press run.

Readers with unusually large bank accounts, or who receive unusually large cheques, may have to alter the formatting line 280. Alternatively, they could reduce their bank balance to a more manageable amount, by sending all excess money to the address at the front of this book!

BANK REC.

```
10 PRINT "OPENING BALANCE"
20 INPUT O
30 PRINT "NUMBER OF CHEQUES"
40 LET A$="CHQ."
50 GOSUB 200
60 PRINT "NUMBER OF PAYMENTS"
70 LET A$="AMT."
80 GOSUB 200
90 PRINT "VALUE OF STANDING ORDERS ETC."
100 LET N=1
110 LET A$="S.O."
120 GOSUB 210
130 PRINT "PRESS S TO SAVE DATA"
140 SCROLL
150 IF INKEY$<>"S" THEN GOTO 150
160 SAVE "BANK REC."
170 CLS
180 GOTO 30
200 INPUT N
210 FOR X=1 TO N
220 IF X/6=INT (X/6) THEN CLS
230 SCROLL
240 PRINTA$;X;
250 INPUT A
260 LET A=A-(2*A AND A$(1)<>"A")
270 LET O=O+A
280 PRINT TAB 8;A,"BALANCE ";O
290 NEXT X
300 PRINT AT 0,0
310 RETURN
```

Credit Cards

At the time of writing, there is some controversy about whether retailers should be allowed to charge a premium on credit card sales. The reason they want to do so is that credit card sales not only cost the retailer money in commission, but also create extra accounting problems. This program will not solve the first problem, but should alleviate the second.

NOTE: This program is part-written for the ZX Printer. This is to provide the user with a hard-copy listing of credit card batches submitted. It is of less value without these listings, but if you don't have a printer, change all LPRINT lines to PRINT, and add a PAUSE after line 100.

Line 50 is the name of the credit card company. A different copy of the program should be used for each company for whom this system applies.

The '96' in line 85 is 100 less the % commission charged by the particular company. Amend this to the actual figure, if necessary.

The ZX Printer provides a tally-roll of submissions (gross and net), which should be compared with statements received from the company.

The screen, meanwhile, displays the NET amount still outstanding.

The write-off routine (lines 130-140) allows for reconciliation in the event of invalid cards being dishonoured by the company.

CREDIT CARDS

```
5 LET T=0
10 LET C=0
15 LET S=C
20 CLS
50 LPRINT "DINERS EXPRESS"
55 PRINT "NO. OF BATCHES"
60 INPUT N
65 FOR J=1 TO N
70 PRINT J
75 INPUT X
80 LET S=S+X
85 LET Y=INT (X*96+.5)/100
90 LET C=C+X-Y
95 LET T=T+Y
100 LPRINT " AMOUNT ";X,"NET ";Y
105 CLS
110 NEXT J
115 PRINT "RECEIPTS"
120 INPUT J
125 LET T=T-J
130 PRINT "WRITE OFF"
135 INPUT J
140 LET T=T-J
145 CLS
150 PRINT"OUTSTANDING: ";T
155 LPRINT "SUBMISSIONS ";S
160 LPRINT "COMMISSION  ";C
165 PRINT AT 5,0;"PRESS S TO SAVE DATA"
170 IF INKEY$<>"S" THEN GOTO 170
500 SAVE "DE"
505 GOTO 10
```

Payroll/Payslip

These two programs perform a similar function, but read the notes carefully to see which one will best suit your particular requirements.

PAYROLL contains the FULL tax tables A, B, C, D, and BR. It is therefore primarily intended for employERs to use in an actual payroll run, in lieu of these tables.

PAYSLIP gives fuller information, including employee's and employer's National Insurance contributions. However, it excludes all the higher rates of tax, and so should NOT be used for or by employees who are liable to pay tax other than at the standard rate. PAYSLIP is primarily intended to enable the vast majority of employEEs to check their own payslips, but may be used, with caution, by many employERs.

Note: NEITHER program is suitable for PREFIX tax codes (D or F).

Subject to the above qualifications, both programs are written in accordance with the Inland Revenue Specifications for Computer Payroll Routines. PAYSLIP also satisfies the D.H.S.S. specifications for N.I. computerised N.I. tables.

The Inland Revenue have given permission for us to print their Test Data (see page 60), so that you can satisfy yourself that the program is working before putting it into practice. (This test data is not applicable to PAYSLIP, as most of the figures quoted refer to the higer tax rates.)

Notes on use.

Three inputs are required. These must all be NUMERIC only.

CODE: Normally enter the numeric element of the code, disregarding the H, L, P, T or V suffix.
Special cases:
Code BR: Enter 0
Code 0T: Enter -1
Code NT: Enter 9999

PAY: Enter Gross Pay to Date (including current week), or current week only if tax code is on week 1 basis.

WEEK: Enter Tax week number (or 1 for week 1 codes).

Outputs:

F/P = Free pay to date
TAX = Tax to date

These outputs are, of course, the same as would be obtained from the manual tax tables.

Notes:

As all the parameters most likely to be changed (e.g. in future budgets etc.)are held outside the body of the program, ammendments are simple.

For monthly pay, simply change the 52 (in lines 70 and 100) to 12, and change the input prompt in line 50.

To economise on variables, T and G and F have been re-used, and no longer have their original (mnemonic) function after line 110.

PAYROLL (DATA)

irst, in immediate mode, establish the following parameters:

```
LET X=5
DIM C(X)
DIM R(X+1)
DIM K(X)
```

Then, still in immediate mode, enter the following data:

TAX DATA FOR 1981/2

CUMULATIVE BANDWIDTHS

```
LET C(1)=11250
LET C(2)=13250
LET C(3)=16750
LET C(4)=22250
LET C(5)=27750
```

TAX RATES

```
LET R(1)=.3
LET R(2)=.4
LET R(3)=.45
LET R(4)=.5
LET R(5)=.55
LET R(6)=.6
```

CUMULATIVE ANNUAL TAXES

```
LET K(1)=3375
LET K(2)=4175
LET K(3)=5750
LET K(4)=8500
LET K(5)=11525
```

Now remember to run by using 'GOTO 0' not 'RUN'

PAYROLL

```
10 PRINT "CODE "
20 INPUT T
30 PRINT "GROSS"
40 INPUT G
50 PRINT "WEEK"
60 INPUT N
70 LET F=N*INT (((T*10+9)/52)*20+1)/20
80 IF T<1 THEN LET F=0
85 CLS
90 PRINT "F/P ";F
100 LET N=N/52
110 IF F>=G THEN GOTO 500
120 LET F=G-F
130 IF F>INT (C(1)*N+1) AND T<>0 THEN GOTO 160
140 LET F=INT F*R(1)
150 GOTO 230
160 LET G=G/G
170 FOR T=2 TO X
180 IF F>INT (C(T)*N+1) THEN LET G=T
190 NEXT T
200 LET F=ABS (INT (C(G)*N*10**4)/10**4-INT F)
210 LET F=F*R(G+1)
220 LET F=INT (K(G)*N*10**4)/10**4+F
230 LET F= INT (F*20+.00001)/20
240 GOTO 510
500 LET F=0
510 PRINT "TAX ";F
```

PAYROLL: TEST DATA

This test data for computerised payroll routines is supplied by the Inland Revenue, and is reprinted with their permission.

WEEK NO.	TAXABLE PAY	RESULTS PER SPEC'N
1	216.00	64.80
5	1083.15	325.00
7	2254.95	773.65
10	3299.56	1144.65
18	7702.95	2942.30
23	12275.00	5098.10
27	6879.55	2167.45
30	4327.50	1298.10
35	8919.80	2810.40
38	8579.00	2609.45
42	17971.55	6865.30
46	5461.90	1638.30
52	27759.69	11530.40

To test a program against this data, use:

CODE: 0T (i.e. '-1')
GROSS: as shown for taxable pay
WEEK: as shown

Results should be EXACTLY as shown.

Notes on use.

In this program, 6 numeric inputs are required.

CODE: as PAYROLL

NI: Enter 1, 2, or 3 for N.I. codes A, B or C.

P.PAY = Previous gross pay (or 0 for wk.1 codes).

PAY: This week's pay only.

WK: as PAYROLL

P.TAX = Previous tax.

Outputs

The output from PAYSLIP is, in fact, a reasonable facsimile of an actual payslip. The left hand column gives the current week's pay details; the right hand column gives cumulative imformation, EXCEPT in the case of 'N.I.', where the figure in the r.h. column is the employER's N.I. contribution for the current week. (EmployEEs using this program to check their payslips may delete all references to variable X.)

Notes:

Unlike PAYROLL, some of the parameters liable to change are embedded in the program (see page 63).

Contracted Out National Insurance rates have NOT been incorporated in this program.

PAYSLIP

```
10 PRINT "CODE "
20 INPUT T
30 PRINT "NI"
40 INPUT I
50 PRINT "P.PAY"
60 INPUT P
70 PRINT "PAY"
80 INPUT W
90 PRINT "WK"
100 INPUT N
110 PRINT "P.TAX"
120 INPUT A
130 CLS
140 LET X=0
150 LET Z=X
160 IF W<27 THEN GOTO 220
170 LET F=200
180 IF W<F THEN LET F=INT (W*2)/2+.25
190 IF W=27 THEN LET F=W
200 LET Z=INT (F*R(I)*100+.49)/100
210 LET X=INT (F*.137*100+.49)/100
220 LET F=N*INT (((T*10+9)/52)*20+1)/20
230 IF T<1 THEN LET F=0
240 LET F=P+W-F
250 LET F=INT (INT F*.3*20+.00001)/20
260 PRINT "GROSS←";W,W+P
270 PRINT "TAX←←←";F-A,F
280 PRINT "N.I.←←";Z,X
290 PRINT TAB 5;"--------"
300 PRINT "NET←←←";W+A-F-Z
```

PAYSLIP DATA

Enter in immediate mode:

```
DIM R(3)
LET R(1)=.0775
LET R(2)=.0275
LET R(3)=0
```

The above parameters are the employee N.I. Rates A, B and C

The following parameters are contained within the program itself:

```
  27 in lines 160,190 = N.I. lower limit
 200 in line  170     = N.I. upper limit
.137 in line  210     = N.I. (employers rate)
  .3 in line  250     = basic tax rate
```

No N.I. test data is supplied: check against tables.

Utilities

A desultory collection of routines which we hope you will find useful.

Byte-Counting

Probably THE most useful routine you will come across whilst working in 1K. Used during program development, GOTO 9999 will reveal exactly the amount of space occupied by your program so far. It won't tell you if your program will actually RUN in 1K, as it measures only the program itself, not the variable or display areas. It's function therefore is in testing various alternative methods of coding, to see which is the most economical. Try it: you may get some surprises, and should find yourself writing 'tighter' code almost immediately.

Line 0

You have seen the expression 'GOTO 0' already in this book, but have you ever actually seen a line number 0? If you can spare the time to key in this 'program', you will! Run and then list the one line and then try to edit or delete your new line number 0! What use is it? Well, apart from amazing your friends and impressing people, none whatever in it's present form. However, if you use POKE 16510,0 in immediate mode, it will always give line number 0 to the FIRST line of your program. (The only reason it is shown as a program line here is because it has no effect if you haven't got at least 1 program line to renumber. Note also, that that line number must originally be less than 256, or you must POKE 16509,0 as well.)

We can think of at least two uses for this:

1: Once you have laboriously entered a machine code routine into a REM statement of a basic program, it can be a little annoying (!!) to accidentally delete this line by a careless entry, so POKE the machine code to line 0 and it is perfectly secure.
2: Why not POKE to line 0 the following:

```
  0 REM COPYRIGHT B. BLOGGS 1982
```

Now your program is safe from piracy by anyone who doesn't know the secret.

BYTE-COUNTING

```
9999 PRINT PEEK 16396+256*PEEK 16397-16562
```

LINE 0

```
255 POKE 16510,0
```

AUTO-RUN

```
9997 STOP
9998 SAVE "PROGRAM NAME"
9999 GOTO 0
```

CHAINS

```
9990 CLS
9991 PRINT "PRESS M TO RETURN TO MENU"
9992 PRINT "OR R TO RE-RUN"
9993 LET I$=INKEY$
9994 IF I$="R" THEN GOTO 0
9995 IF I$="M" THEN LOAD "MENU"
9996 GOTO 9993
```

You have already seen variations of the next 3 routines in the DATA FILES section. If you were not quite sure what was happening, read on.

Auto-Run

Auto-run is a feature of the ZX81 which is not available even on many more expensive computers! If you append these 3 lines to any of your programs, and then SAVE by entering GOTO 9998, subsequent LOADings will cause the program to auto-run. If you are very short of space, omit line 9997 (if your program stops anyway); reduce the program name to a single character; change line 9999 to 'RUN'.

Chains

This is a sort of 'auto-load', which, if used in conjunction with AUTO-RUN, will enable you to simulate quite long programs on your 1K ZX81, by chaining any number of shorter programs together. You can either load each program on a tape directly from the previous one (no need in that case for a program name in line 9995) or, as in the example, end each program with an option to re-run or recall a common menu, from which any other program may be loaded. (In this case, you must either remember to rewind the cassette after each loading, or make a cassette on which each program, and the menu, is recorded several times.)

Menu

A menu, in computer terms, as otherwise, is a list of options available. It is a luxury which few 1K programs can afford, but an EXTERNAL menu, like this one which offers the first three programs in the book, used in conjunction with CHAINS and AUTO-RUN, can give a very professional touch to your program cassettes.

MENU

```
5 LET A$="CHINESE HOROSCOPE"
10 LET B$="NUMEROLOGY"
15 LET C$="I CHING"
20 PRINT AT 0,2;"FORTUNE TELLING<<<<<BY V+H"
25 PRINT AT 5,0;"1:";A$
30 PRINT AT 7,0;"2:";B$
35 PRINT AT 9,0;"3:";C$
40 PRINT AT 17,6;"ENTER YOUR SELECTION"
45 LET I$=INKEY$
50 IF I$<"1" OR I$>"3" THEN GOTO 45
55 IF I$="1" THEN LOAD A$
60 IF I$="2" THEN LOAD B$
65 IF I$="3" THEN LOAD C$
```

Machine Code Loader

A machine code loader can be as simple or as complex as you like. This one has been deliberately made as simple as possible, so that it can be keyed-in in a matter of seconds, rather than loaded from tape every time it is required. You can, of course, add as many frills as you desire once you become seriously involved in machine code programming.

To Run:

Amend line 1 so that there are the required number of X's. (Actually any character will do.)

Amend line 10 so that the loop is executed the required number of times. i.e. Replace LIMIT with 16514 PLUS no. of X's MINUS 1

Run, entering your code 2 hex digits at a time.

SAVE ! (Most important, because if the program crashes you will have to switch off and start again.)

Run the m/c program by executing the USR function. The most foolproof method is:

```
LET DUMMY=USR 16514
```

although PRINT USR 16514 or GOTO USR 16514 may be appropriate on occasions.

If you make a mistake during entry, it is usually advisable to start again. Once you get used to machine code, however, you will find that you can edit many of the bytes directly from the 'garbage' which appears on screen when you LIST line 1.

MACHINE CODE LOADER

```
1 REM XXXXX (AS MANY AS BYTES IN M/C PROGRAM)
10 FOR A=16514 TO LIMIT
15 INPUT H$
20 POKE A,(CODE H$(1)-28)*16+CODE H$(2)-28
25 NEXT A
```

Line Renumbering

It is a short step from 'Line 0', where we renumbered the first line of a program, to a complete routine which renumbers an entire program. This is very useful if you want your programs to look neat and tidy, and essential if you want to sell or publish them. It is also a handy routine to have loaded if you ever find that you have left insufficient room between line numbers to add a later amendment.

Notes

As listed, the routine will renumber in steps of 5 (S in line 8960), starting with 5 (F in line 8965). You may alter these values as you wish.

The routine works by scanning the Program Area (addresses 16509 onwards) looking for the Newline character (118). When this is found, the following 2 bytes must contain the next line number, so the new value is POKEd in (lines 8980-8985).

Line 9005 prevents the routine renumbering itself, by causing a break (list) when line number 8960 is encountered. Note, however, that the FIRST new line number is always POKEd in immediately, without the program scan, so the routine WILL renumber it's own first line if you do not have a program entered for it to act on.

To run, enter GOTO 8960 (or RUN 8960) at any time. When the listing appears (eventually!), you must amend all GOTO and GOSUB commands to suit the new line numbers, and finally delete the routine.

The major problems with this routine are that it is very slow, and it takes up a lot of room itself, so can be used only on very short programs. (Also it takes no less than 65 keystrokes to delete it.) There is a way round these problems, of course...use Machine Code...

LINE RENUMBERING

```
8960 LET S=5
8965 LET F=S
8970 LET A=16509
8975 LET P=INT (F/256)
8980 POKE A,P
8985 POKE A+1,F-(P*256)
8990 LET D=PEEK A
8995 LET A=A+1
9000 IF D<>118 THEN GOTO 8990
9005 IF PEEK A=35 THEN GOTO 9020
9010 LET F=F+S
9015 GOTO 8975
9020 LIST
```

If you are going to enter this routine using the loader on page 69, put 27 X's in line 1 and set LIMIT to 16540.

Important: Because this routine is translated directly from the previous Basic program, it still searches for line number 8960 (byte 16h). Therefore, call by entering the basic line:

```
8960 LET D=USR 16514
```

Do not call in immediate mode, unless you insert a dummy line 8960 (e.g. REM). (If you can't remember that, 9000 will do just as well, as only the high order byte of the line number is tested.)

Notice that bytes 01 and 02 commence renumbering at address 16542, and not 16509 as in the Basic program. This is because the routine itself is the first program line; if you make any changes, therefore, remember to set these bytes to the start of the SECOND program line.

M/C RENUMBERING

```
00    21 9E 40  START  :LD HL,16542;1ST LINE
03    11 00 00         LD DE,0     ;LINE # 0
06    06 05     RENUMB:LD B,5      ;STEPS OF 5
08    13        STEP   :INC DE     ;NEW NO.IN DE
09    10 FD            DJNZ 253    ;TO STEP
0B    72               LD(HL)D     ;  INSERT
0C    23               INC HL      ;  NEW
0D    73               LD(HL),E    ;  LINE NO.
0E    23        NXTBYT:INC HL      ;NEXT BYTE
0F    7E               LD A,(HL)   ;
10    FE 76            CP 118      ;IS IT N/L
12    20 FA            JRNZ 250    ;IF NOT,NXTBYT
14    23               INC HL      ;NEXT LINE NO.
15    7E               LD A,(HL)   ;
16    FE 23            CP 35       ;IS IT 8960
18    20 EC            JRNZ 236    ;IF NOT,RENUMB
1A    C9               RET         ;RETURN
```

Games

At last, the section you have all been waiting for. Don't get to excited, though; it is our belief that the best source of computer games is in your own head, and in any case you will almost certainly get more enjoyment out of writing games than actually playing them. What we will do in this section, then, is show you six of the many TYPES of computer game, with an example of each. After that, if you want to discover the next craze to replace Space Invaders, it is up to you.

Play Your Cards

Before you rack your brains thinking up new games, search the attic and the local toyshop for the hundreds of existing games just crying out for computerisation. Many have already been done (If you haven't written a Mastermind program yet, you must be the only computer programmer who hasn't, so stop reading immediately and get on with it!), but there are many more waiting to be 'discovered'.

This program bears a remarkable resemblance to a well known TV show. (Yes, that's right, Crossroads.) All you have to do is guess whether the next card laid will be higher or lower than the last one. As soon as you make a wrong guess, the computer takes its turn, and the winner is whoever has the greatest number of correct guesses. Simple but addictive: the definition of a good computer game.

The computer's strategy, such as it is, is to make the logical choice, except for the mid-card 8, in which case it chooses randomly (line 170).

Lines 535-540 are yet another pause loop. This time the pause is greater when the computer is playing (pseudo-thinking time) as N<>0.

PLAY YOUR CARDS

```
10 REM 23456789TJQKA
20 LET X=0
30 LET N=X
40 LET A$=""
50 GOSUB 500
60 PRINT AT 1,N;"HIGHER OR LOWER"
70 LET I$=INKEY$
80 IF I$<>"H" AND I$<>"L" THEN GOTO 70
90 GOSUB 500
100 GOSUB 400
110 IF A THEN GOTO 70
120 PRINT AT 1,N;"YOUR SCORE←←←←";LEN A$-2
130 LET A$=""
140 LET N=2
150 GOSUB 500
160 LET I$="LOWER←"
170 IF X+RND<6.5 THEN LET I$="HIGHER"
180 PRINT I$
190 GOSUB 535
200 GOSUB 500
210 GOSUB 400
220 IF A THEN GOTO 160
230 PRINT "MY SCORE←←←←←←";LEN A$-2
240 GOSUB 535
250 CLS
260 RUN
400 LET A=1
410 IF (I$(1)="H" AND X<=Y) OR (I$(1)=
        "L" AND X>=Y) THEN LET A=A-A
420 RETURN
500 LET Y=X
510 LET X=INT (RND*13)
520 LET A$=A$+CHR$ PEEK (16514+X)
530 PRINT AT N,0;A$
535 FOR A=1 TO N*30+20
540 NEXT A
550 RETURN
```

Headlines

Computers are excellent for playing word games. Another program which you MUST write for yourself is Hangman, and for the more ambitious there are the 'pseudo-conversational' programs like the famous Eliza. (Did anyone else watch the BBC TV series 'Adventure Game', in which an Apple computer was programmed with an Eliza variant, and served no function except to delay contestants by engaging them in pointless conversation? Very few people actually sussed it out.)

Headlines is a variation of another computer classic. It prints endless and often amusing rubbish onto the screen. Obviously you can modify the strings as you like; obscene versions are fun, but definitely not for publication!

Note

The subroutine at line 90 is not the most efficient way of selecting items from a string, but was chosen because it prints slowly, letter by letter, and also it allows the items to be of varying length.

HEADLINES

```
5 LET A$="←TOPLESS←MAD←NUDE←BORING
          ←SEXY←DEAD←FAT←UGLY←"
10 GOSUB 90
15 LET A$="←POTATO←MADMAN←POP-STAR←
          ACTRESS←TEACHER←VICAR←CAT←PRINCE←"
20 GOSUB 90
25 PRINT "←IN";
30 LET A$="←STRIKE←SEX←PEACE←DEATH
          ←RACE←SHOCK←BLACK←WHITE←"
35 GOSUB 90
40 LET A$="←DRAMA←WAR←RUMOUR←RIOT←
          MOVIE←HORROR←SCANDAL←PLEDGE←"
45 GOSUB 90
50 FOR J=1 TO 20
55 NEXT J
60 CLS
65 RUN
90 LET X=INT (RND*8)
95 FOR J=1 TO LEN A$
100 IF A$(J)="←" THEN GOTO 110
105 NEXT J
110 LET X=X-1
115 IF X>-1 THEN GOTO 105
120 PRINT A$(J);
125 IF A$(J+1)="←" THEN RETURN
130 LET J=J+1
135 GOTO 120
```

Royalty Cheque

This is our contribution to the growing range of educational games. (Ugh! Turn immediately to Bowls! Alternatively, don't TELL anyone it is supposed to be educational.)

You have just written a wonderful computer program to control the world's economy, solve the mysteries of the universe, or, better still, play chess in 1K! Naturally you send it off to 'Your World of Practical Printing Out Today', who agree to publish it and pay you so much per published page. You are very pleased when you see that you program occupies a complete page, but it has been mixed in with a number of advertisments, so to work out your commission you must calculate the proportion of the page which actually contains your listing.

The 'text' represents your program, and naturally the ads. have been blanked out. You have two attempts to work out your payment. The first attempt tests your spatial recognition as well as your maths, as you must estimate (or count, until you get good at this) the ratio of program:ads.

If your first attempt is wrong, you will be told the number of ads., and also whether your guess was high or low, so the second attempt is pure mental arithmetic. Remember, however, that you must be accurate to the nearest penny.

Note: Line 25 does work, because RND takes a different value each time it is called. This is a very economical 'random branch', as it uses no numbers or variables, and it tends to produce a ROUGHLY equal division of text/blanks.

Incidentally, there is about one chance in 10 billion that this program actually will produce a valid machine code program to control the world's economy. What would the authors of 'The Last One' have to say about that ?!

ROYALTY CHEQUE

```
5 LET Z=0
10 FOR J=0 TO 14
15 FOR K=0 TO 9
20 PRINT AT J,K;
25 IF RND>RND THEN GOTO 45
30 PRINT "←"
35 LET Z=Z+1
40 GOTO 50
45 PRINT CHR$ (RND*16+28)
50 NEXT K
55 NEXT J
60 LET K=INT (RND*10)+1
65 PRINT
70 PRINT "£";K;"/PAGE"
75 INPUT A
80 LET J=K-K*Z/150
85 LET J=INT (J*100+.5)/100
90 IF A=J THEN GOTO 150
95 LET A$="TOO LOW←."
100 IF A>J THEN LET A$(5 TO 8)="HIGH"
105 PRINT A$;" THERE ARE ";Z;" ADS."
110 INPUT A
115 IF A=J THEN GOTO 150
120 PRINT "NO, £";J
125 CLEAR
130 INPUT A$
135 CLS
140 RUN
150 PRINT "YES"
155 GOTO 125
```

Moving Graphics

The following three games all adopt a different solution to the problem of fitting a meaningful screen display, a moving graphics routine and a sensible program, all into 1K.

In BOWLS, the playing area is restricted to just the top 10 lines of the screen, which makes perfect sense in the context of the game, and leaves plenty of memory free for everything else. CATCH has possibly the least impressive screen display since you first turned the ZX81 on! But, having only two characters on screen does mean that they can wander over the entire screen area without any problem. SPACE RESCUE fills (almost) the entire screen with characters, and they all move! How? By using the very concise and economical ZX81 command SCROLL.

Bowls

This game is played in much the same way as the 'real' version. Two players (you can be both of them if you like) take turns rolling their bowls towards the jack. ,The object is not to hit it, but to land closest to it. Player 1 takes bowls 1 and 3; player 2 takes 2 and 4. The winner of each 'end' bowls first in the next end. The ZX81 places the jack for you, to prevent the first player having any advantage.

If you hit any other bowl, or the jack, this is dispatched to the gutter and out of play. Clearly this can often be a good thing to do to your opponent's bowls, but if the jack is out of play, the end must be re-played, so try to avoid deliberately hitting the jack except as a last resort. Your own bowls will land in the gutter if they stray off the sides of the green (which is 10 lanes wide: there is one lane on each side of the 8 marked mats), or off the end. (Any roll of dist: greater than 30)

Three inputs are required to set up each bowl:
Mat no: In the game as listed, mat numbers are deleted as they are used to discourage continuous bowling from the same mat. However, you still CAN bowl from any mat (or indeed from mats 0 or 9, which are not marked), and can easily modify the program to leave mat numbers on screen.

BOWLS

```
5 LET Z=0
10 FOR J=Z TO 8
15 PRINT J
20 NEXT J
25 PRINT AT RND*8,RND*10+20;"."
30 FOR K=Z TO PI
35 PRINT AT Z,Z;"MAT←"
40 INPUT L
45 LET A$=CHR$(K+157)
50 PRINT AT L,Z;A$
55 PRINT AT Z,Z;"BIAS"
60 INPUT B
65 IF ABS B>5 OR NOT B THEN GOTO 60
70 PRINT AT Z,Z;"DIST"
75 INPUT S
80 FOR J=Z TO S
85 PRINT AT L,J;"←"
90 IF J>.7*S THEN LET L=B/5+L
95 IF L<Z OR L>9 OR J=31 THEN GOTO 110
100 PRINT AT L,J+1;A$
105 NEXT J
110 NEXT K
115 INPUT B
120 CLS
125 RUN
```

Bias: This is the most interesting feature of the game. The bowls do not roll in a straight line! Enter any number from minus 5 (severe left bias) to 5 (severe right bias). Obviously an entry of zero will not be accepted. (line 65)
Dist: You must judge for yourself how far to roll the bowl, and also the extent to which the distance of the roll affects the bias. Remember that a distance of 30 will reach the end of the screen. Entering 'dist' also sets the bowl rolling. After all 4 bowls have been bowled, enter any NUMERIC value for the next end.

Notes.

The number zero is used a lot in this program: Hence a variable has been set to this value in line 5 to conserve memory.

The moving graphics routine is in lines 85 (to delete the previous bowl position) and 100 (to print the new position).

A$ (CHR$(K+157)) is the inverse numbered bowl (1-4).

Line 95 deals harshly with all bowls which stray out of range. What problems arise without this line? (There are 3 possibilities: see if you can predict all 3, and then try it.)

Catch

This is the archetypical moving graphics game. You can elaborate on the basic routine as much as you like, to produce a vast variety of games. In this version, the $ characters moves randomly about the screen, and you must try to catch it in the 'jaws' of your character, the letter C. As has become almost a standard for ZX81 games, the keys 5-8 have been re-defined to control direction of movement (lines 65 & 75). The POKEs in lines 25 & 30 start the timer, and the PEEKs in line 100 read it. The PEEKs in line 85 check the current PRINT position (which is immediately to the right of the C, if you haven't forgotten the semi-colon in line 40) to see if there is a $ character there. Lines 50, 60, 70 and 80 prevent the characters straying off screen. Notice that, as there are only 2 characters on screen, the CLS function is used to avoid 'overprinting' the characters with blanks, as in the previous program.

CATCH

```
5 LET A=11
10 LET B=15
15 LET X=A
20 LET Y=20
25 POKE 16436,250
30 POKE 16437,255
35 PRINT AT A,B;"$"
40 PRINT AT X,Y;"C";
45 LET A=A+1-2*(RND<.5)
50 LET A=A+(A<0)-(A>21)
55 LET B=B+1-2*(RND<.5)
60 LET B=B+(B<0)-(B>31)
65 LET X=X+(INKEY$="6")-(INKEY$="7")
70 LET X=X+(X<0)-(X>21)
75 LET Y=Y+(INKEY$="8")-(INKEY$="5")
80 LET Y=Y+(Y<0)-(Y>31)
85 IF PEEK (PEEK 16398+256* PEEK 16399)=
        13 THEN GOTO 100
90 CLS
95 GOTO 35
100 PRINT AT 0,0;INT (1310.6-(PEEK 16435
      +256*PEEK 16437)/50);"SECONDS"
105 CLEAR
110 INPUT X$
115 CLS
120 RUN
```

Space Rescue

Well, we had to include a space game somewhere! In this one, you have to rescue the captives (.) from their guards (*). You score 1 point for each rescue, but lose 5 points every time you run into a guard (at which point the action pauses momentarily for your score to be displayed). By using SCROLL, the only character which actually has to be moved by the program is your ship (X), see lines 20, 25 and 35. (Note that there is no routine to prevent you crashing off screen.) In this program, line 40 sets the PRINT position immediately BELOW the X, and line 45 looks at what is in this position, as before.

The string in line 3 is the arrangement of captives and guards, and you can alter this to make the game easier or harder, as you like. We found the arrangement as shown hard enough, and rarely managed to obtain a positive score! Try omitting the inner 2 asterisks at first.

SPACE RESCUE

```
1 LET Y=15
2 LET S=0
3 PRINT AT 17,RND*30;"**.**"
20 PRINT AT 9,Y;"←"
25 LET Y=Y+(INKEY$="8")-(INKEY$="5")
30 SCROLL
35 PRINT AT 9,Y;"X"
40 PRINT TAB Y;
45 LET U=PEEK (PEEK 16398+256*PEEK 16399)
50 LET S=S+(U=27)
55 IF U<>23 THEN GOTO 3
60 LET S=S-5
65 PRINT S
70 GOTO 3
```

Bibliography

The following is a list of all the books (which we know about) covering the Sinclair ZX81.

As most of the books are from small publishers, there may be more which we haven't seen (Publishers please drop us a line).

We have been unable to verify the latest prices of some of the books, so apologise for any errors.

All books dealing SPECIFICALLY with the 1K ZX81 have been marked with an asterisk.

Publisher	Title	Price
V&H	*What Can I Do With 1K?	4.95
	What Can I Do With 16K?	4.95
Melbourne House	*Not Only 30 Programs	6.95
	Understanding Your ZX81 ROM	8.95
	Machine Lang.Made Simple	8.95
Phipps Assocs.	ZX 81 Pocketbook	5.95
Linsac	ZX 81 Companion	7.95
	Monitor Listing	4.50
Newnes	ZX81 Basic Book	4.95
Timedata	ZX80/81 Magic Book	4.75
	Explorers Guide to ZX81	4.95
Comp. pubns.	Making The Most Of Your ZX81	6.95
	Stretching ZX81 to the Limits	6.95
Interface	Gateway Guide to the ZX81	5.95
	Master Machine Code (ZX80/1)	5.95
	Getting Acquainted with ZX81	4.95
	*34 Amazing Games	3.95
	49 Explosive Games	5.25
Database	50 Rip-roaring Games (ZX80/1)	4.95
R.Frances	*Cambridge Collection	4.95
Hewson	Hints & Tips	4.25
MacMillan	ZX81 Prog'ing for Real Applicns.	7.95
Sigma	*Byteing Deeper into Your ZX81	4.95
Shiva	*Peek, Poke, Byte & RAM	4.95
Cambridge U.P.	ZX Programmers Comp.	5.00
B.B.C.	30 Hour Basic (ZX81)	5.50

COMPUTER DATING QUESTIONNAIRE

NAME .....

SEX (M/F)

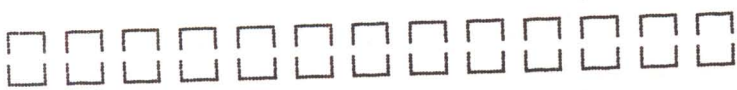
Pick the word/phrase from each of the following pairs which most accurately describes yourself.

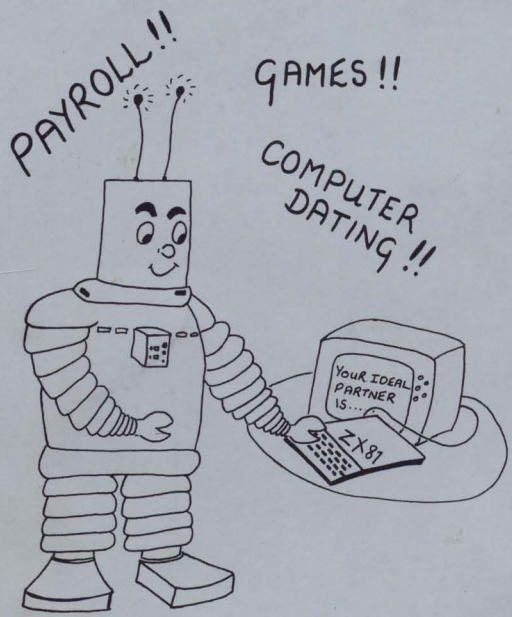
SHY/EXTROVERT
FUN LOVING/HUMOURLESS
INTELLECTUAL/THICK AS TWO SHORT PLANKS
FASHION CONSCIOUS/DRESS LIKE A TRAMP
CAREFREE/CAUTIOUS
SOCIABLE/LONER

Mark 3 of the following with a tick if they are your interests, AND 3 with a cross if you are definitely NOT interested.

A	POP MUSIC	J	COMPUTERS
B	CLASSICAL MUSIC	K	POLITICS (LEFTISH)
C	WATCHING SPORT	L	POLITICS (RIGHTISH)
D	PLAYING SPORT	M	POLITICS (GENERAL)
E	CINEMA	N	TRAVEL
F	THEATRE	O	ARTS/CRAFTS
G	DISCO	P	POETRY
H	FOOD (RESTAURANT)	Q	ANIMALS
I	FOOD (COOKING)	R	READING

For official use only




PAYROLL!!
GAMES!!
COMPUTER DATING!!
YOUR IDEAL PARTNER IS...
ZX81